Workflow and Direct Communication in the openDIEL

Nick Moran (UTK), Tanner Curren (Maryville College)
Mentor: Kwai Wong (UTK)

About openDIEL

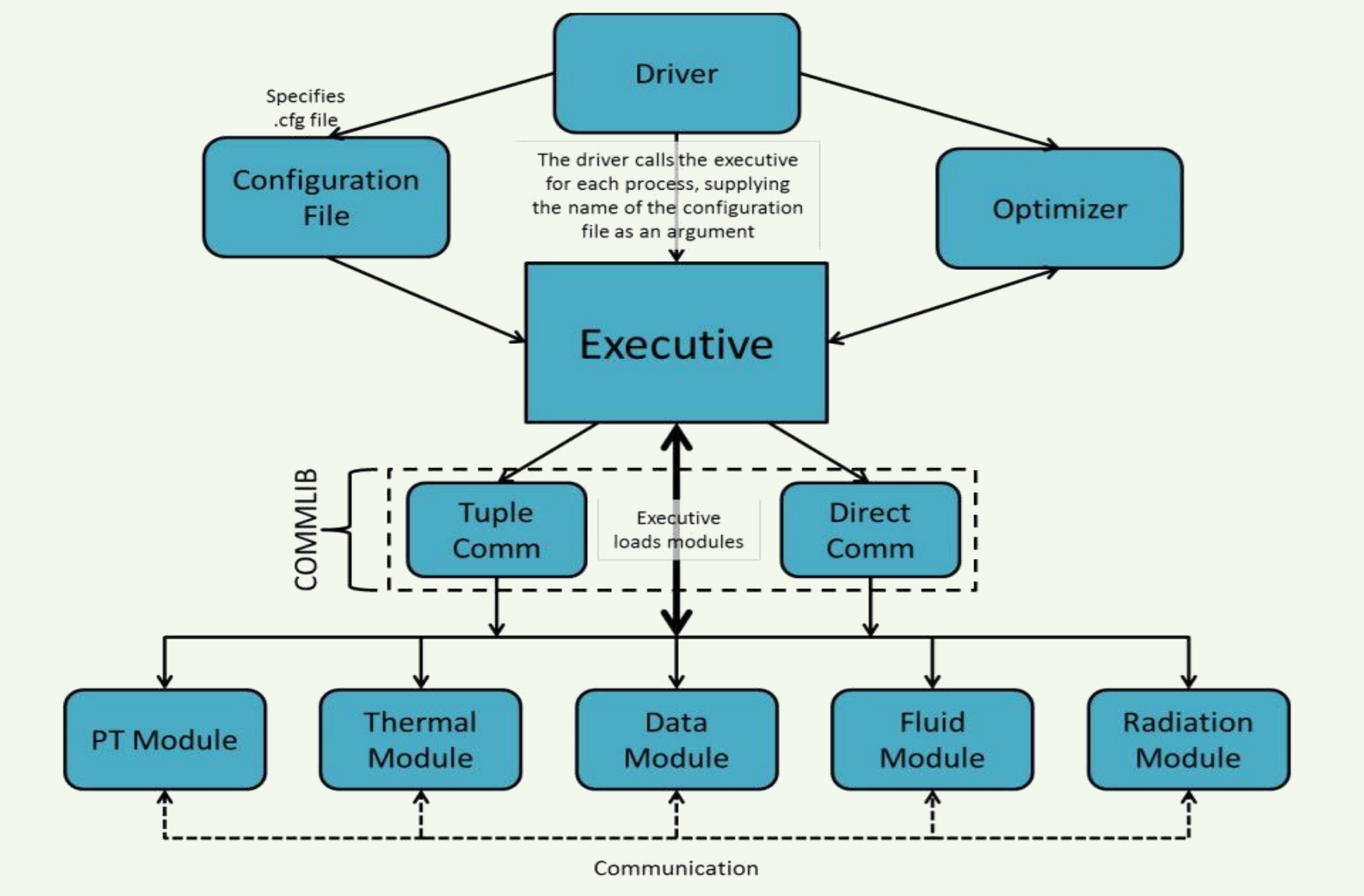
The **openDIEL** (open Distributive Interoperable Executive Library) aims to facilitate communication between **user-created loosely coupled simulations**.

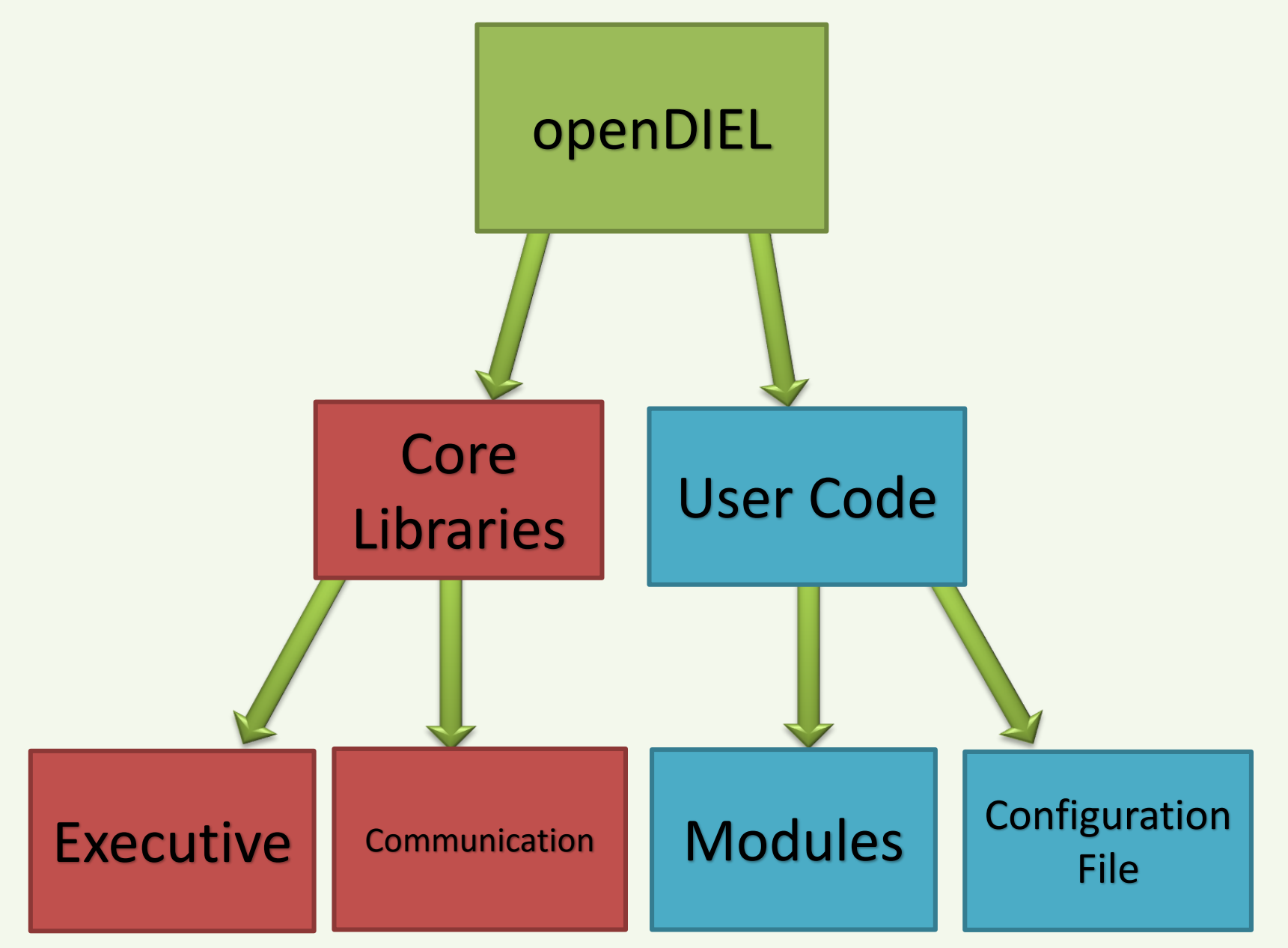
Loosely coupled simulations are mostly serial programs that rely on data points from other simulations; these simulations get their input from and send output to other simulations.

The openDIEL communicates through a tuple server-based method, and now is able to communicate large chunks of contiguous memory more efficiently through **direct communication**.

The organization of these simulations is organized and facilitated by the **workflow** implementation of the openDIEL.

openDIEL Organization





User Interaction with openDIEL

```
shared_bc_sizes = []
tuple_space_size=0
```

Global Options

Specification of shared boundary condition array sizes and the tuple space size (number of processes)

shared_bc_sizes: array of integers describing the sizes of the shared boundary condition arrays.
tuple_space_size: number of processes the tuple space uses.

```
modules=(
    {
    function="modep7"
    args=()
    libtype="static"
    library="libmodep7.a"
    splitdir="ep7-rv-workflow"
    size=512
    points=()
    },
    {
    function="modreadvars"
    args=("in.rvi", "monthly")
    libtype="static"
    splitdir="ep7-rv-workflow"
    library="libmodreadvars.a"
    size=512
    points=()
    },
  {
    function="RAnalysis"
    args=()
    libtype="static"
    library="libRAnalysis.a"
    splitdir="ep7-rv-r-workflow"
    size=1
    points=()
  }
)
```

Module Specific Options

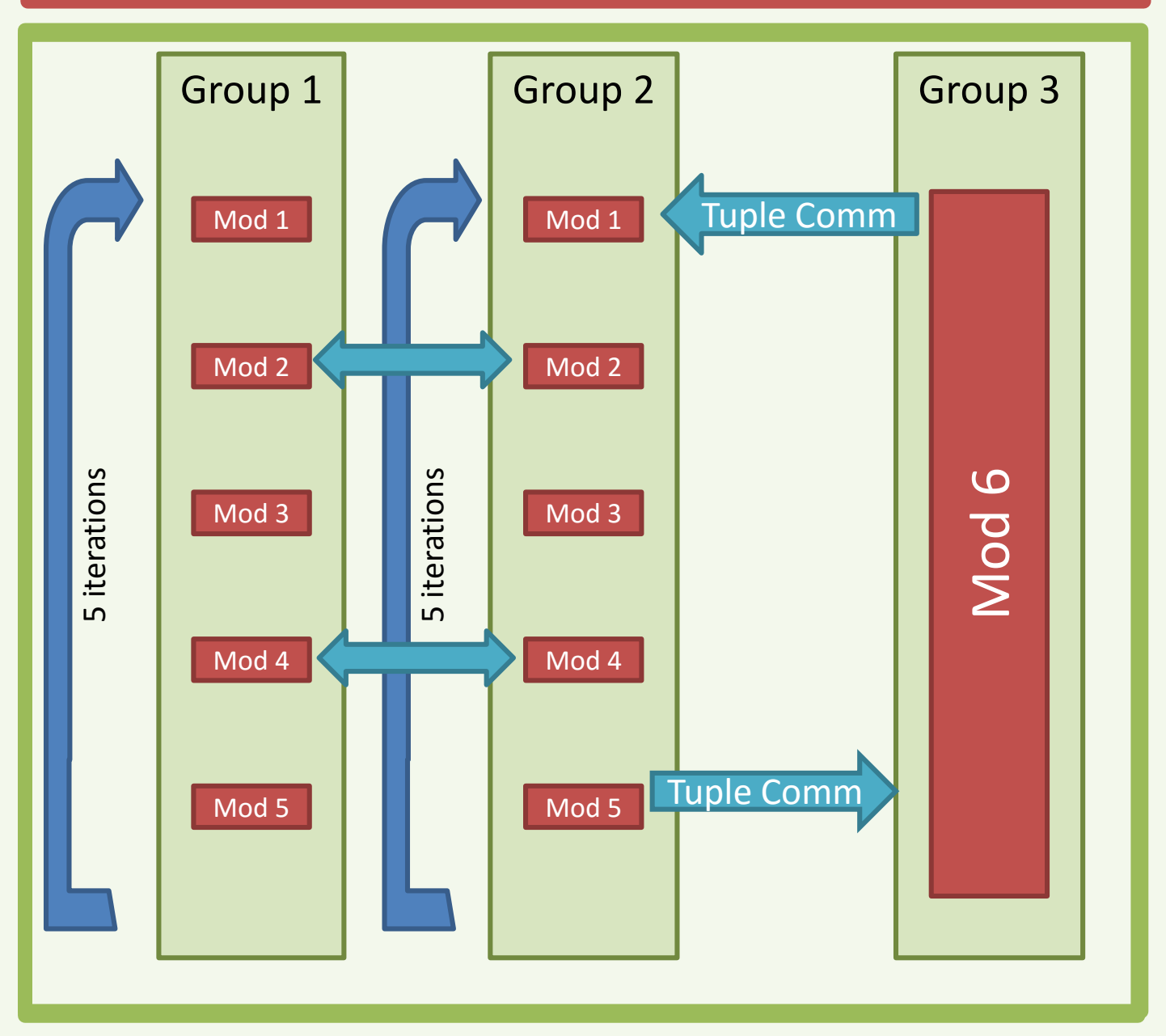
Provided here is information specific to each module, such as identifiers, the amount of processes each module requires, and any arguments that should be passed.

function: Name of module main function
args: Arguments to pass to the function
libtype: Shared libraries will be opened with dlopen, static libraries will be previously linked
libname: If shared, this is the file from which to look up the symbol named in "function"
size: number of processes this module should use
points: describes which SBCs this module may access
splitdir: base name of the directory that the module will run in.

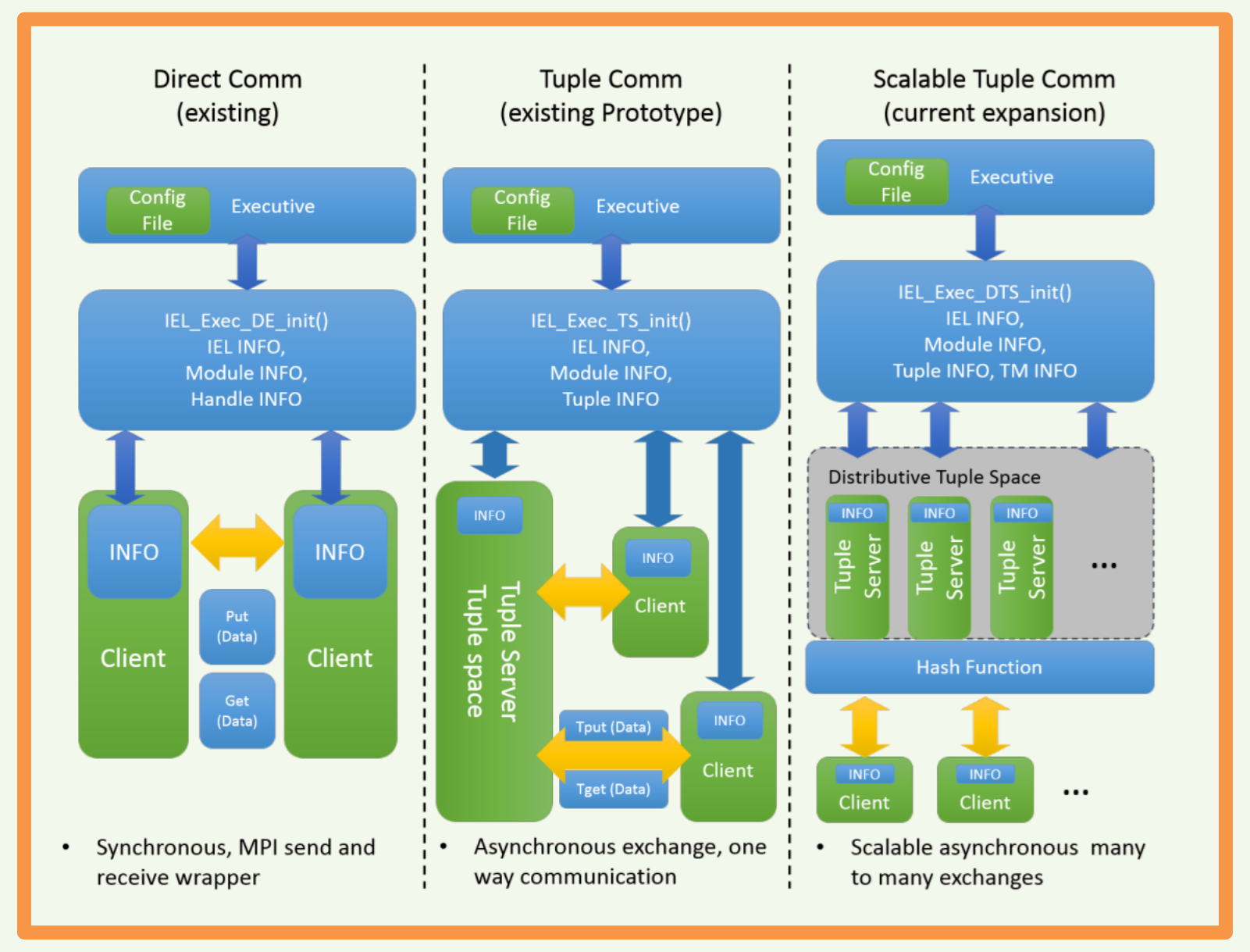
```
workflow:
{

  groups:
  {
    ep7-readvars:
    {
      order=("modep7", "modreadvars")
    }
    RAnalysis:
    {
      order=("RAnalysis")
    }
  }
```

Workflow Specification

Description of the workflow to be executed.



The openDIEL provides two methods of communication between processes: Direct communication allows processes to share points on an array, while the Tuple Server acts as a place to queue messages based on tags for other modules to retrieve.


Direct Comm (existing)
Tuple Comm (existing Prototype)
Scalable Tuple Comm (current expansion)
Config File
Executive
IEL_Exec_DE_init() IEL INFO, Module INFO, Handle INFO
IEL_Exec_TS_init() IEL INFO, Module INFO, Tuple INFO
IEL_Exec_DTS_init() IEL INFO, Module INFO, Tuple INFO, TM INFO
INFO
Client
Put (Data)
Get (Data)
Tuple Server
Tuple space
Tput (Data)
Tget (Data)
Distributive Tuple Space
Hash Function
Synchronous, MPI send and receive wrapper
Asynchronous exchange, one way communication
Scalable asynchronous many to many exchanges

The openDIEL workflow engine allows users to organize different parts of their workflow into module "groups". Each group is a set of modules that may operate sequentially and repeatedly. Modules can communicate with each other across groups using the Tuple Server or Direct Communication

Direct Communication

- Wrappers around MPI to communicate large chunks of data
- Functions **IEL_send** (nonblocking), **IEL_recv** (blocking) and **IEL_move** (blocking)
- Uses a set of **shared boundary conditions** in a conceptual grid, each process having access to a different section of this grid
- Movement parameters specify point-to-point transfer

Configuration file using direct communication:

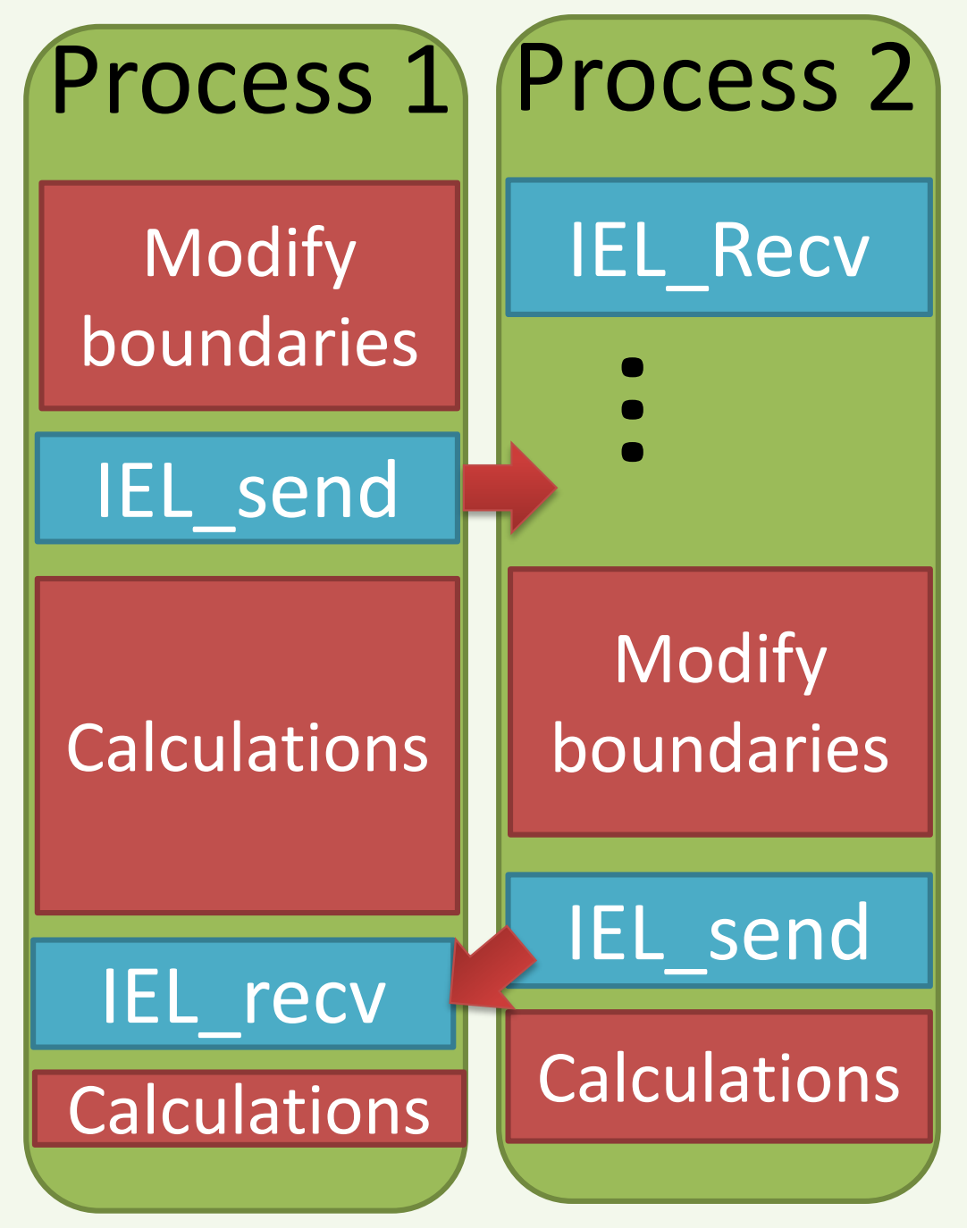
```
modules = (
{
    function = "jacobi";
    args = ();
    libtype = "static";
    library = "libjacobi.a";
    size = 4;
    points = (
        (([0,16]),([0,16]),([0,0]),([0,0]),([0,0]),
         ([0,0])),

        (([0,16]),([0,16]),([0,16]),([0,16]),([0,0]),
         ([0,0])),

        (([0,0]),([0,0]),([0,16]),([0,16]),([0,16]),
         ([0,16])),

        (([0,0]),([0,0]),([0,0]),([0,0]),([0,16]),
         ([0,16]))
    );
}
)

movement = (
    (0, ([0, 16]), (0), (1)),
    (1, ([0, 16]), (1), (0)),
    (2, ([0, 16]), (1), (2)),
    (3, ([0, 16]), (2), (1)),
    (4, ([0, 16]), (2), (3)),
    (5, ([0, 16]), (3), (2))
)
```

Example MPI Communication Code

```
    if( mype < npes-1 )
/* Send my data down to the processor below me; Only npes-1 do this */
     MPI_Send(&t[NRL][1], NC, MPI_FLOAT, mype+1, DOWN, MPI_COMM_WORLD);
    if( mype != 0 )
/* Sending my data up to the processor above me ; Only npes-1 do this  */
     MPI_Send(&t[1][1], NC, MPI_FLOAT, mype-1, UP, MPI_COMM_WORLD);
    if( mype != 0 )
/*  Receive new data from UP processor of any source */
     MPI_Recv(&t[0][1], NC, MPI_FLOAT, MPI_ANY_SOURCE, DOWN,
              MPI_COMM_WORLD, &status);
    if( mype != npes-1 )
/*  Receive new data from DOWN processsor of any source */
     MPI_Recv(&t[NRL+1][1], NC, MPI_FLOAT, MPI_ANY_SOURCE, UP,
              MPI_COMM_WORLD, &status);
```

Example IEL Communication Code

```
    // Exchange data with the process below
    if (myRank != NPES - 1)
      IEL_move(exec_info, myRank + 1);
    // Exchange data with the process above
    if (myRank != 0)
      IEL_move(exec_info, myRank - 1);
```



Workflow Use Case: Energy Plus

EnergyPlus – (Single Program Multiple Data)
EnergyPlus is a program developed by the DOE to model the power consumption of buildings. The openDIEL workflow engine is used to run EP7, Readvars, and an R analysis script to perform statistical analysis on the dataset as a whole.

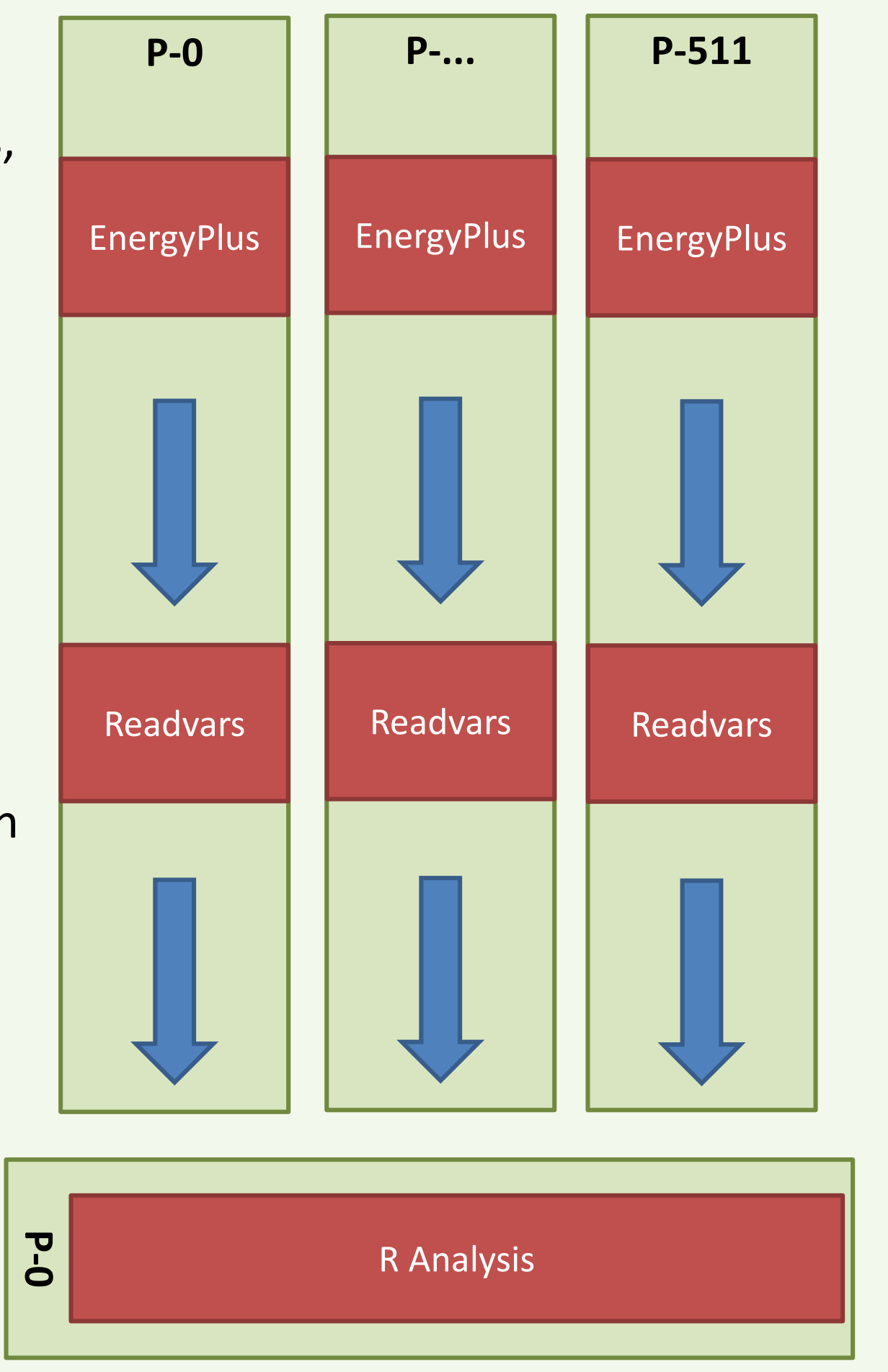
```
workflow:
{

  groups:
  {
    ep7-readvars:
    {
      order=("modep7", "modreadvars")
    }
    RAnalysis:
    {
      order=("RAnalysis")
    }
  }
```

One group is used for this case, containing EnergyPlus, Readvars, and R Analysis (a module that runs arbitrary R scripts).

Each EP7/Readvars module executes in a separate directory on a different set of input files.

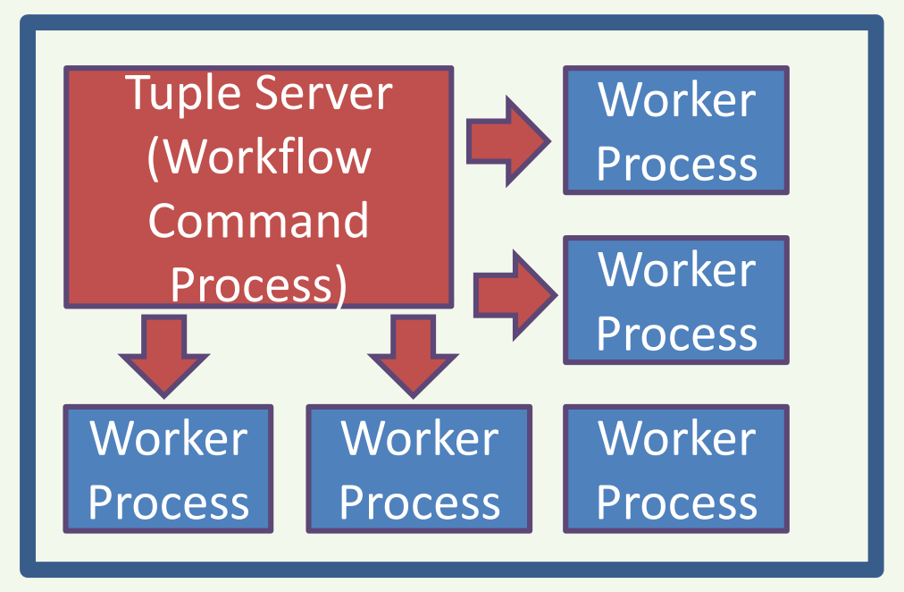
Once EP7 is complete, Readvars extracts the relevant information from the output files.

After this data is collected, an R script is used to perform statistical analysis on the produced data.



Future Work

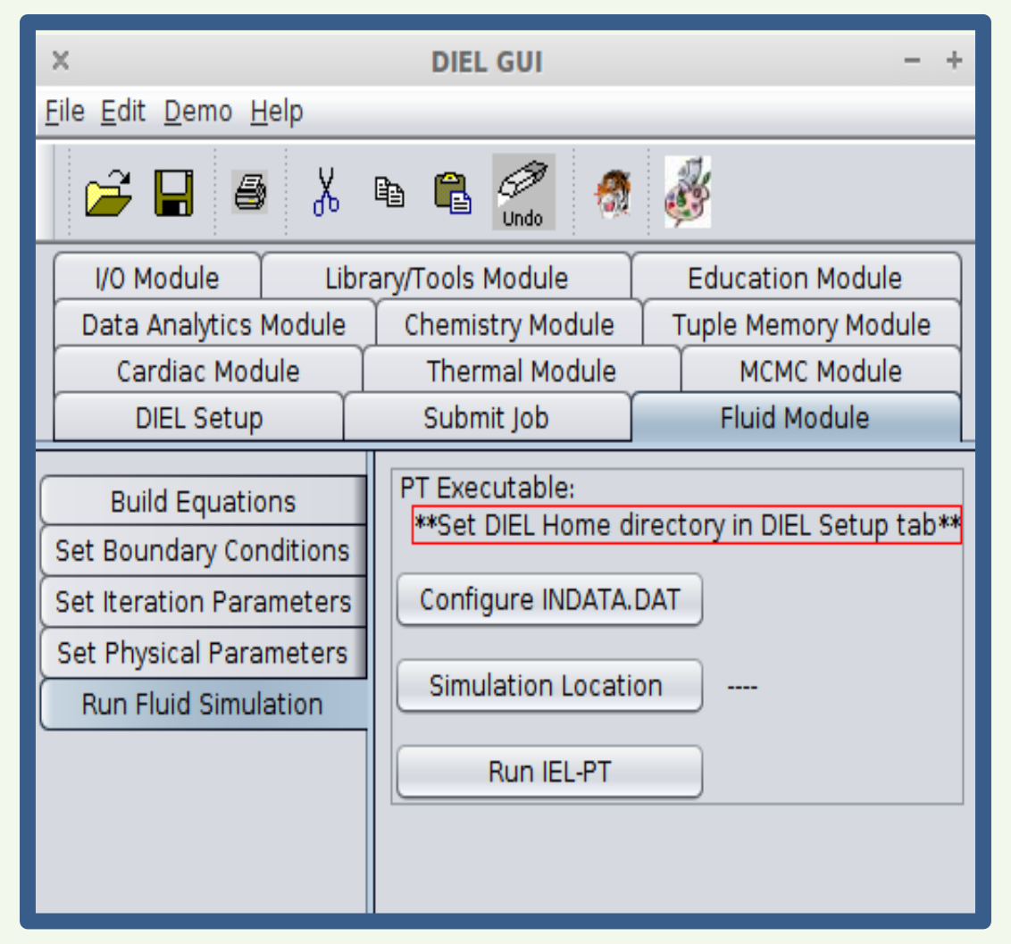
Workflow/Tuple Server Control Module
The tuple server will be expanded to read a makefile-like list of dependencies and dynamically schedule tasks using the available processes.



ModMaker
The ModMaker toolset will be ported from bash to python and made available on the Python Package Index (PyPi) for an easy, one line installation.

Tuple Server Scaling
The tuple server will be expanded to allow multiple processes to operate as a single server (currently each processes operates separately).

openDIEL GUI
The openDIEL GUI will be extended to expose openDIEL features in a more user friendly package.



PT Executable:
Set DIEL Home directory in DIEL Setup tab
Simulation Location ----

Acknowledgments

Mentorship
Kwai Wong

Special Thanks
NSF: Funding
ORNL: Hosting
JICS: Compute Resources, Workspace